1

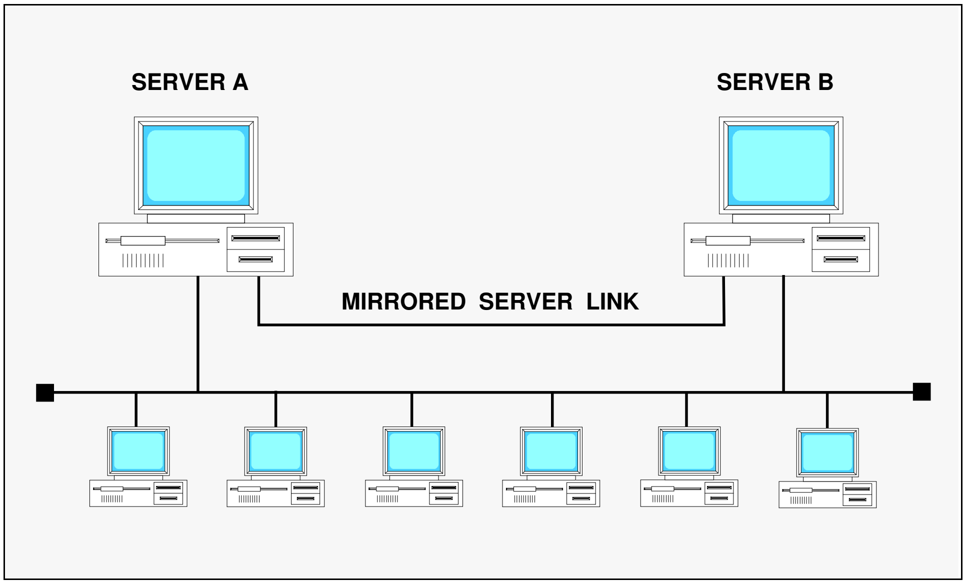
NetWare SFT III Mirrored Servers

Introduction

This document defines the Mirrored Server Link (MSL) driver interface for NetWare SFT Level III. The server-to-server communication is an integral element of the SFT III scheme.

SFT III is a method of mirroring the state of a NetWare v3.x server on a secondary server. The primary and secondary servers, connected by a high speed, low latency communications link, function together as one logical server. The resulting entity, the *SFT III Server*, is resilient to most single hardware failures. The SFT III system transparently mirrors all server application NLMs that do not directly access hardware, including database applications and other file service protocols.

Only the primary server appears to clients on the internetwork as a server. The secondary server maintains an event-by-event mirror of the primary server's activities. If the primary server fails or the console operator downs the server to add or upgrade hardware, the secondary server will become the active primary component of the logical server. Clients using mirrored servers see no loss of state or service (other than a slight pause as the LAN communications protocol changes its route to get to the active server). The result is uninterrupted service to the client.



1.1 Mirrored Servers

The Mirrored Server Link

SFT III operates with regular NetWare-compatible servers and uses normal NetWare LAN and Disk drivers. All synchronizing and mirroring functions are implemented entirely in software. However, a dedicated communications link, the Mirrored Server Link (MSL), is required to relay information between the mirrored servers.

MSL Communications

The MSL driver is responsible for the transmission and reception of the following information through the Mirrored Server Link.

- Message Packets
- Message Acknowledgements
- HoldOff Notifications
- Emergency Notifications

The Mirrored Server Link employs a point-to-point message passing protocol that requires message acknowledgement. The primary information communicated through the link is in the form of *message packets*. These messages relay the information required to keep the servers mirrored. Chapter 2 describes the general format of a message packet and provides an overview of the driver's required message handling functions.

The MSL driver is responsible for message data integrity. When a message packet is received, the driver must ensure that it is error free and must return a *message acknowledgement* to the other server. The driver then informs the operating system of the received message. The operating system may respond by accepting the message, rejecting the message, or placing the message on hold for redelivery at a later time. In this last case, the driver must send a *holdoff notification* to the other server. The holdoff notification essentially informs the other server to extend the timeout on any messages that have not yet received an acknowledgement.

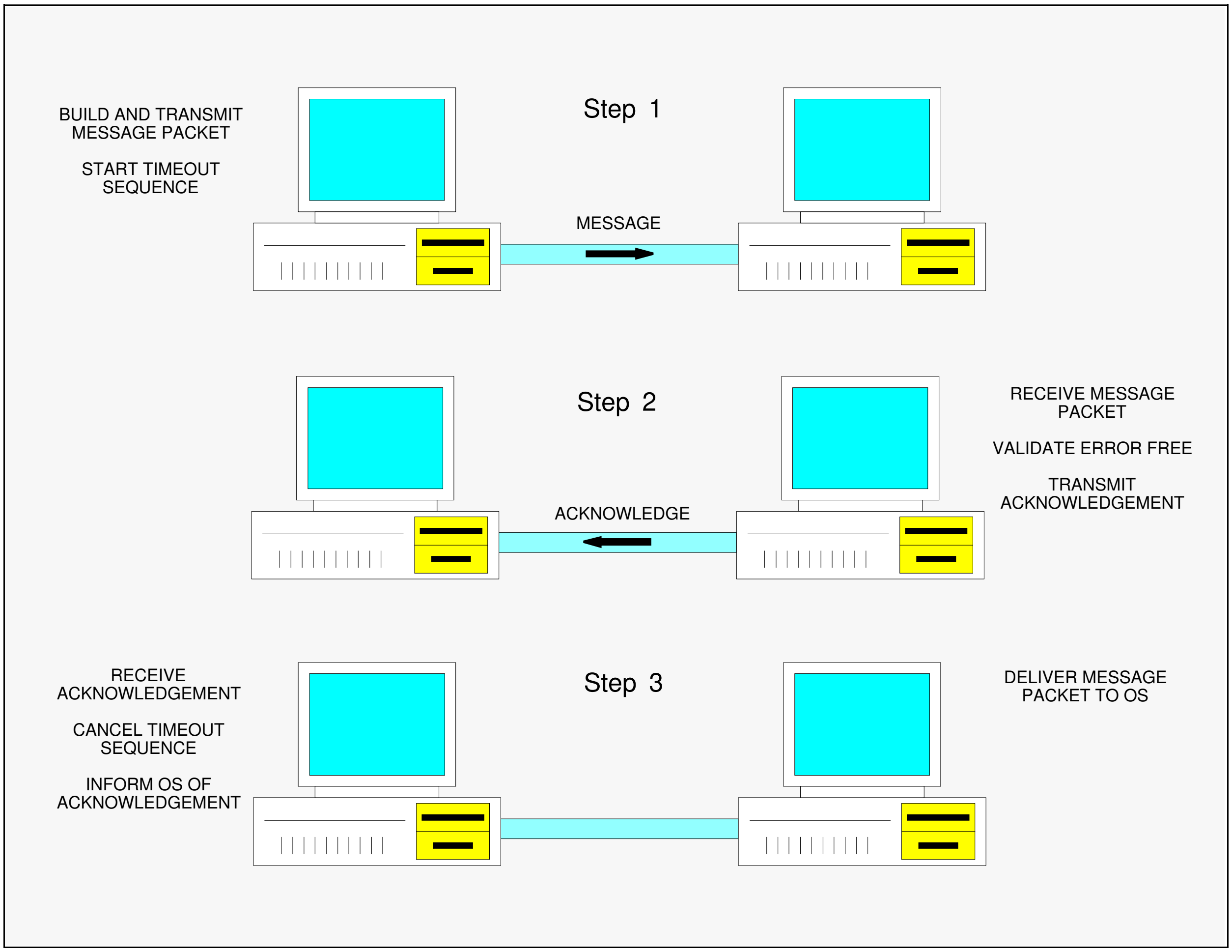
Whenever a server is about to go down, the SFT III operating system has the MSL driver transmit an *emergency notification* to the other server. This emergency signal basically notifies the other server that the sending server is going down.

Note: *Messages, acknowledgements,* and *notifications* may be sent from either server at any given time. With NetWare SFT III, mirrored server communications is bidirectional and is not always from the primary to the secondary server.

Basic Message Sequence

The figure below illustrates the basic server-to-server communication managed by the Mirrored Server Link driver. The link in the diagram can be a single communications channel or can include a supplementary channel used for acknowledgments and emergency notifications of server failure. The supplementary channel, if used, would bypass the adapter's message receive buffer.

Chapter 4 of this document provides a complete description of the driver procedures used for handling message transmission and reception.



1.2 Basic Message Sequence

MSL Design Criteria

Criteria for the Mirrored Server Link include:

- High Speed
- High Volume / Bandwidth
- Low CPU Utilization
- Low Latency
- Capacity for Large Message Size

With the exception of FDDI adapters, most regular LAN adapters lack the bandwidth needed for the Mirrored Server Link. Another drawback of using normal LAN adapters for the MSL is the overhead involved (address decoding, access control, etc...). An MSL adapter requires only point-to-point communications. The simplest MSL solution would be a high-speed fully duplexed UART.

MSL Speed

The speed of the Mirrored Server Link communication is crucial. The server re-synchronizing and re-mirroring functions, as well as overall server performance, will suffer significantly if the MSL is too slow. 100 Mb/sec should be considered a minimum acceptable criterion.

Volume/Bandwidth

The volume of MSL message traffic will be at least two times the number of LAN packets sent and received by the server. For example, an NCP File Read from cache that receives and sends one LAN packet will generate 4 MSL messages (3 of which will be 24 bytes long).

For good performance, the MSL should provide a minimum bandwidth equal to the total bandwidth of all LAN adapters and disk controllers in the server. All disk writes must go over the MSL, as well as all LAN packets received by the server. However, disk reads and LAN packets sent to clients do not go over the MSL.

CPU Utilization

Low impact on the CPU is an important characteristic of the high-speed link. The adapter must be capable of being set up to send and receive message data with minimal CPU utilization. This implies using 32-bit paths, zero or minimal wait states on transfers to and from the adapter, and simple adapter setup to start transmits and receive messages. The adapter should have interrupts to minimize service latency.

Since most packets are small, adapters that require extensive setup time may not be the best solution. One recommended implementation would employ fast 32-bit wide shared memory for transmit and receive buffers.

Latency

The Mirrored Server Link requires low latency. The primary server can perform some processing while transmitting a message packet, but at some point the server must synchronize by waiting until the other server acknowledges the message.

Latency pertains not only to the transmission itself, but to the *whole transmission process*: downloading the message to the adapter, transmitting the message, and receiving the acknowledgement. Low latency is crucial in all stages.

There is one mechanism used to reduce the impact of latency. As messages get backlogged, the MSL driver can send multiple messages within one physical packet. The result is more work performed per latency hit. However, this strategy is still not sufficient for high latency communication channels. For example, the high latency of SCSI handshaking in most implementations makes it a bad choice for the MSL.

Message Size

The Mirrored Server Link will typically be given many short (24 byte) messages and occasionally large messages. The minimum message size is 24 bytes. The adapter must be able to send at least 400 byte messages. The capability to handle message sizes up to 4K+24 bytes (and any needed headers) would be about the optimum. The MSL driver specifies its maximum message size using the variable, *MaximumCommDriverDataLength*.

Hardware Error Checking

Hardware should do a CRC check on the data if possible. If designing new hardware specifically for use as an SFT III MSL adapter, error checking at the hardware level is a significant advantage.

Other considerations

The data within a message packet determines the message destination in system memory. The adapter should have a receive buffer with space for queuing several received messages. This avoids copying message data twice. It would also be useful to queue several messages for transmission. To obtain maximum throughput, the adapter should allow for loading and unloading messages while it is transmitting or receiving other messages.

The SFT III synchronization algorithm requires an acknowledgement indicating that the other server has received the message. Adapters without a hardware-level acknowledgement mechanism must generate some sort of acknowledgement in the data stream. This acknowledgement may be "piggy-backed" on a data packet. With Token-Ring adapters, setting the *received bit* on the message should be sufficient acknowledgement.

One method for handling acknowledgement packets efficiently employs a secondary communications channel that bypasses the normal message path. If the driver has multiple messages queued in the receive buffer, a second channel could provide a quicker acknowledgement response. The channel should be at least as fast as the main message channel.

Dual Mirrored Server Links

***Note**: The initial release of NetWare SFT III will not support the dual MSL feature. Novell's implementation of support for dual MSL adapters may differ from the outline below. This section is for information purposes only.*

With a single link between the mirrored servers, SFT III can guarantee uninterrupted service. However, problems occur when the MSL adapter fails on the *primary* server. In this case, service would continue but there would be no way to re-mirror the servers without bringing down the primary server to fix the MSL adapter.

To eliminate this type of re-mirroring problem, each server would need *two* MSL adapters: a primary adapter and a standby adapter. The standby adapter is activated when the primary MSL adapter fails. The system would communicate through the standby link only until the secondary server fully mirrors the primary server. The console operator could then bring down the primary server to replace the faulty adapter without loss of service to the client.

The primary and standby MSL adapters are not required to be the same. A normal LAN adapter can function temporarily as the standby MSL adapter by writing a special driver for the board. The system uses the standby link only until the secondary server can takeover. In addition, the dual system would require only a single physical link, since the console operator could switch cables between adapters without bringing down the primary server.

This is only one example of configuring an SFT III Server to handle potential failure in the MSL adapters. Other configurations may include two identical adapters in each server and redundant cabling between them. The key to exploiting these possibilities lies in the MSL driver.